## AMENDMENTS TO THE CLAIMS

1. (Previously Presented) A method for storing data in a repository comprising the steps of:

storing in a first table data for one or more default attributes of a first object type, wherein a default attribute is an attribute included in a default object type, used by an application;

storing in a second table, separate from said first table, data for one or more default attributes of a second object type used by said application; and

storing in a third table, separate from said first and second tables, data for a first custom attribute of said first object type, wherein a custom attribute is an attribute that is added to a default object type, and data for a second custom attribute of said second object type, wherein said first custom attribute and said second custom attribute have the same data type; and

upgrading said application, wherein upgrading said application comprises the steps of:

processing the data stored in said first table, wherein processing comprises:

creating a first replacement table to hold the data from said first table;

copying the data from said first table to said first replacement table, wherein data from said one or more default attributes of said first object type is copied from said first table into said first replacement table; and

deleting said first table;

processing the data stored in said second table, wherein processing comprises:

creating a second replacement table to hold the data from said second table;

copying the data from said second table to said second replacement table, wherein data from said one or more default attributes of said second object type is copied from said second table to said second replacement table; and

deleting said second table; and

retaining, in said third table, values for said first custom attribute of said first object type and said second custom attribute of said second object type.

2. (original) The method of Claim 1, wherein:

said third table includes

at least one instance-identifying column, wherein each row of said third table stores in said at least one instance-identifying column data that uniquely identifies an object instance that is associated with the row;

at least one attribute-identifying column, wherein each row of said third table stores in said at least one attribute-identifying column data that identifies a custom attribute of the object instance that is associated with the row; and

at least one value column, wherein each row of said third table stores in said at least one value column one or more values for the custom attribute that is identified in said at least one attribute-identifying column; and

said at least one value column of said third table stores data that has the same data type as said first custom attribute and said second custom attribute.

3. (canceled)

4. (original) The method of Claim 1, further comprising the step of retrieving an object instance of said first object type, wherein said object instance of said first object type includes data from said third table associated with said first custom attribute of said first object type.

5. (original) The method of Claim 1, wherein:

said third table stores values for custom attributes of a plurality of object types including said first object type and said second object type; and the method further comprises assigning to every instance of every object type of said plurality of object types an instance identifier value that is unique relative to every instance of every object type of said plurality of object types.

6. (Currently Amended) A method for storing data in a repository comprising the steps of:

storing in a first table data for one or more default attributes of a first object type, wherein a default attribute is an attribute included in a default object type, used by an application;

storing in a second table, separate from said first table, data for one or more default attributes of a second object type used by said application; and

storing in a third table, separate from said first and second tables, data for a first custom attribute of said first object type, wherein a custom attribute is an attribute that is added to a default object type, and data for a second custom attribute of said second object type, wherein said first custom attribute and said second custom attribute have the same data type; and

storing, in a catalog table, data defining said first custom attribute of said first object type and said second custom attribute of said second object type; and

wherein said catalog table stores data that identifies custom attributes for at least one object type;

the method further comprises performing the following steps in response to a request to access an object instance of a particular object type:

reading said catalog table to determine the custom attributes of said particular object type; and

based on [[the]] information from said catalog table, constructing in volatile memory data structures that indicate the custom attributes of said particular object type; and in response to a subsequent request to access [[an]] a different object instance of said particular object type, inspecting said data structures, without accessing said catalog table, to determine the custom attributes of said particular object type.

7. (original) The method of Claim 6, wherein said catalog table includes:

at least one first catalog column, wherein each row of said catalog table stores, within said at least one first catalog column, data that identifies an object type associated with the row,

at least one second catalog column, wherein each row of said catalog table stores, within said at least one second catalog column, data that identifies a custom attribute of the object type that is associated with the row, and

at least one third catalog column, wherein each row of said catalog table stores, within said at least one third catalog column, data identifying a data type of the custom attribute that is identified in said second catalog column.

8. (original) The method of Claim 7, the method further comprising the step of retrieving, in response to a request for an object instance of said first object type, the value of said first custom attribute associated with said object instance by performing the steps of:

determining the data type of said first custom attribute from said third catalog column of a catalog-table row stored in said catalog table, wherein:

data in said at least one first catalog column of said catalog-table row matches data identifying said first object type; and

data in said at least one second catalog column of said catalog-table row matches data identifying said first custom attribute;

based on the data type of said first custom attribute, determining the identity of said third table; and retrieving, from a value column of a third-table row stored in said third table, the value of said first custom attribute, wherein:

data uniquely identifying said object instance matches data in at least one instance-identifying column of said third-table row; and

data identifying said first custom attribute matches data in at least one attribute-identifying column of said third-table row.

9. (original) The method of Claim 7, further comprising the step of storing in said catalog table data defining a custom object type, separate from said first object type and said second object type, wherein the step of storing said custom object type includes the step of inserting a row into said catalog table, wherein said row includes:

within said at least one first catalog column, data that identifies said custom object type;

within said at least one second catalog column, data that identifies a custom attribute of said custom object type, and

within said at least one third catalog column, data identifying a data type of said custom attribute that is identified in said at least one second catalog column.

10. (Currently Amended) A method for storing data in a repository comprising the steps of:

storing in a first table data for one or more default attributes of a first object type, wherein a default attribute is an attribute included in a default object type, used by an application;

storing in a second table, separate from said first table, data for one or more default attributes of a second object type used by said application; and

storing in a third table, separate from said first and second tables, data for a first custom attribute of said first object type, wherein a custom attribute is an attribute that is added to a default object type, and data for a second custom attribute of said second object type, wherein said first custom attribute and said second custom attribute have the same data type; and

storing, in a catalog table, data defining said first custom attribute of said first object type and said second custom attribute of said second object type; and

wherein said catalog table stores data that identifies custom attributes for a plurality of object types;

the method further comprises performing the following steps in response to said application being launched:

reading said catalog table to determine custom attributes from said plurality of object types; and

based on [[the]] information from said catalog table, constructing in volatile memory data structures that indicate the custom attributes of each of said plurality of object types; and

in response to a request to access a different object instance of a particular object type of said plurality of object types, inspecting said data structures, without accessing said catalog table, to determine the custom attributes of said particular object type.

11. (canceled)

12. (Previously Presented) A method for storing data in a repository comprising the steps of:

storing in a first table data for one or more default attributes of an object type used by an application;

storing in a second table, separate from said first table, data for a first custom attribute of said object type that is of a first data type; and storing in a third table, separate from said first and second tables, data for a second custom attribute of said object type that is of a second data type, wherein said first custom attribute and said second custom attribute have different data types; and

upgrading said application, wherein upgrading said application comprises the steps of:

processing the data stored in said first table, wherein processing comprises:

creating a first replacement table to hold the data from said first table;

copying the data from said first table to said first replacement table, wherein data from said one or more default attributes of said object type is copied from said first table into said first replacement table; and

deleting said first table;

retaining, in said second table, values for said first custom attribute of said object type; and

retaining, in said third table, values for said second custom attribute of said object type.

13. (original) The method of Claim 12, wherein:

said object type is a first object type of a plurality of object types used by said application; and

the method further includes:

storing in a fourth table, separate from said first, second and third tables, data for one or more default attributes of a second object type of said plurality of object types; and

storing in said second table data for a third custom attribute of said second object type, wherein said third custom attribute of said second object type is of said first data type.

14. (original) The method of Claim 12, wherein:

said second table includes

at least one instance-identifying column, wherein each row of said second table stores in said at least one instance-identifying column data that uniquely identifies an object instance that is associated with the row;

at least one attribute-identifying column, wherein each row of said second table stores in said at least one attribute-identifying column data that identifies a custom attribute of the object instance that is associated with the row; and

at least one value column, wherein each row of said second table stores in said at least one value column one or more values for the custom attribute that is identified in said at least one attribute-identifying column; and

said at least one value column of said second table stores data that has the same data type as said first custom attribute.

15. (canceled)

16. (original) The method of Claim 12, further comprising the step of retrieving an object instance of said object type, wherein said object instance of said object type includes data from said second table associated with said first custom attribute of said object type.

17. (original) The method of Claim 12, wherein:

said second table stores values for custom attributes of a plurality of object types including said object type, wherein the custom attributes are of said first data type;

said third table stores values for custom attributes of a plurality of object types including said object type, wherein the custom attributes are of said second data type; and

the method further comprises assigning to every instance of every object type of said plurality of object types an instance identifier value that is unique relative to every instance of every object type of said plurality of object types.

18. (Currently Amended) A method for storing data in a repository comprising the steps of:

storing in a first table data for one or more default attributes of an object type used by an application;

storing in a second table, separate from said first table, data for a first custom attribute of said object type that is of a first data type; and

storing in a third table, separate from said first and second tables, data for a second custom attribute of said object type that is of a second data type, wherein said first custom attribute and said second custom attribute have different data types; and

storing, in a catalog table, data defining said first custom attribute of said object type and said second custom attribute of said object type; and

wherein said catalog table stores data that identifies custom attributes for at least one object type;

the method further comprises performing the following steps in response to a request to access an object instance of a particular object type:

reading said catalog table to determine the custom attributes of said particular object type; and

based on [[the]] information from said catalog table, constructing in volatile memory data structures that indicate the custom attributes of said particular object type; and

in response to a subsequent request to access an object instance of said particular object type, inspecting said data structures, without accessing said catalog table, to determine the custom attributes of said particular object type.

19. (original) The method of Claim 18, wherein said catalog table includes:

at least one first catalog column, wherein each row of said catalog table stores, within said at least one first catalog column, data that identifies an object type associated with the row, at least one second catalog column, wherein each row of said catalog table stores, within said at least one second catalog column, data that identifies a custom attribute of the object type that is associated with the row, and

at least one third catalog column, wherein each row of said catalog table stores, within said at least one third catalog column, data identifying a data type of the custom attribute that is identified in said second catalog column.

20. (original) The method of Claim 19, the method further comprising the step of retrieving, in response to a request for an object instance of said object type, the value of said first custom attribute associated with said object instance by performing the steps of:

determining the data type of said first custom attribute from said third catalog column of a catalog-table row stored in said catalog table, wherein:

data in said at least one first catalog column of said catalog-table row matches data identifying said object type; and

data in said at least one second catalog column of said catalog-table row matches data identifying said first custom attribute;

based on the data type of said first custom attribute, determining the identity of said second table; and

retrieving, from a value column of a second-table row stored in said second table, the value of said first custom attribute, wherein:

data uniquely identifying said object instance matches data in at least one instance-identifying column of said second-table row; and

data identifying said first custom attribute matches data in at least one attribute-identifying column of said second-table row.

21. (original) The method of Claim 19, further comprising the step of storing in said catalog table data defining a custom object type, separate from said object type, wherein the step of storing said custom object type includes the step of inserting a row into said catalog table, wherein the row includes:

within said at least one first catalog column, data that identifies said custom object type;

within said at least one second catalog column, data that identifies a custom attribute of said custom object type, and

within said at least one third catalog column, data identifying a data type of said custom attribute that is identified in said at least one second catalog column.

22. (Currently Amended) A method for storing data in a repository comprising the steps of:

storing in a first table data for one or more default attributes of an object type used by an application;

storing in a second table, separate from said first table, data for a first custom attribute of said object type that is of a first data type; and

storing in a third table, separate from said first and second tables, data for a second custom attribute of said object type that is of a second data type, wherein said first custom attribute and said second custom attribute have different data types; and

storing, in a catalog table, data defining said first custom attribute of said object type and said second custom attribute of said object type; and wherein

said catalog table stores data that identifies custom attributes for a plurality of object types;

the method further comprises performing the following steps in response to said application being launched:

reading said catalog table to determine custom attributes from said plurality of object types; and based on [[the]] information from said catalog table, constructing in volatile memory data structures that indicate the custom attributes of each of said plurality of object types; and

in response to a request to access an object instance of a particular object type of said plurality of object types, inspecting said data structures, without accessing said catalog table, to determine the custom attributes of said particular object type.

23-29. (canceled)

30. (Previously Presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 1.

31. (Previously Presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 2.

32. (canceled)

33. (Previously Presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 4.

34. (Previously Presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 5.

35. (Previously Presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 6.

36. (Previously Presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 7.

37. (Previously Presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 8.

38. (Previously Presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 9.

39. (Previously Presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 10.

40. (canceled)

41. (Previously Presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 12.

42. (Previously Presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 13.

43. (Previously Presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 14.

44. (canceled)

45. (Previously Presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 16.

46. (Previously Presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 17.

47. (Previously Presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 18.

48. (Previously Presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 19.

49. (Previously Presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 20.

50. (Previously Presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 21.

51. (Previously Presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 22.

52. (canceled)